PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 : G06F 17/60 | A2

(11) International Publication Number: WO 00/65506

(43) International Publication Date: 2 November 2000 (02.11.00)

(21) International Application Number: PCT/US00/10686

(22) International Filing Date: 21 April 2000 (21.04.00)

(30) Priority Data:

09/295,577	22 April 1999 (22.04.99)	US
Not furnished	20 April 2000 (20.04.00)	US

(71) Applicant: MS INTERGATE, INC. [US/US]; 5100 East La Palma Avenue, Anaheim Hills, CA 92807 (US).

(72) Inventors: HALAVAIS, Richard, A.; 5100 East La Palma Avenue, Anaheim Hills, CA 92807 (US). CHUNG, Tony, Cheng-Tong; 2035 Viento Verano Drive, Diamond Bar, CA 91765 (US).

(74) Agents: COESTER, Thomas, M. et al.; Blakely, Sokoloff, Taylor & Zafman, 7th floor, 12400 Wilshire Boulevard, Los Angeles, CA 90025-1026 (US).

(81) Designated States: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: INDIVIDUAL SEAT SELECTION TICKETING AND RESERVATION SYSTEM

(57) Abstract

A method and system of providing distributed access to a venue reservation system. A seating chart for the requested venue is served to an unafilliated client node. The client tentatively selects a seat from the seating chart. Upon verification of the clients' payment information, the selected seat is designated as reserved on subsequent servings of the seating chart.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav	TM	Turkmenistan
BF	Burkina Faso	GR	Greece		Republic of Macedonia	TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's	NZ	New Zealand		
CM	Cameroon		Republic of Korea	PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

INDIVIDUAL SEAT SELECTION TICKETING AND RESERVATION SYSTEM

BACKGROUND

This is a continutation-in-part of Serial No. 09/295,577, filed on April 22, 1999, entitled INDIVIDUAL SEAT SELECTION TICKETING AND RESERVATION SYSTEM.

(1) Field of the Invention

The invention relates to making reservations over a wide-area network. More specifically, the invention relates to permitting a widely distributed reservation system accessible over a wide-area network without dedicated hardware and software.

(2) Background

Various ticketing and reservation systems are disclosed in the prior art. In U. S. Patent 5,797,126 (1988), Helbling, et al., describes a series of individual kiosks in wireless communication with a central station where a visitor can locate events of interest, view an excerpt of scenes from that venue and purchase tickets. This requires users to physically visit a remote site to avail themselves of the service. Additionally, said prior art makes extensive use of what is called "kiosks," thereby requiring specialized hardware and software to render the services.

U.S. Patent 4,974,252 describes a more interactive theater attendance system where patrons are permitted two way communications between themselves and a broadcast center but this system requires that persons be in attendance at the theater and, further, some attendant be present at the remote broadcast center.

Other prior art exists in the reservations arena, but typically fails to cure the shortcomings of the art set forth above.

BRIEF SUMMARY OF THE INVENTION

A method and system of providing distributed access to a venue reservation system is disclosed. A seating chart for the requested venue is served to an unafilliated client node. The client tentatively selects a seat from the seating chart. Upon verification of the clients' payment information, the selected seat is designated as reserved on subsequent servings of the seating chart.

BRIEF DESCRIPTION OF THE DRAWINGS

The advantages of the present invention will be more fully understood hereinafter as a result of a detailed description of a preferred embodiment when taken in conjunction with the following drawings in which:

Figure 1 is a flow diagram of the present invention illustrating the major components thereof and the interactivity that takes place between the potential customer and the instant invention.

Figure 2 is an illustration of the concept of the present invention utilizing the internet as the Wide Area Network to which users connect to perform the desired function and shows an example of a remotely located user accessing the functionality of the instant invention for purposes of reserving seats for a dinner theater performance in a distant city.

Figure 3 is a illustration of the concept of the present invention refined down to the functionality of reserving specific seats and blocking from duplicate sale those seat that are already reserved.

Figure 4 is a sample of the screens seen by a remote user of the instant invention during a session wherein the user selects and orders 4 specific seats for a distant dinner theater show.

Figure 5 is a complete code set for one exemplary embodiment of the present invention.

Figure 6 is a flow diagram of a host server operation in response to a venue request in an alternative embodiment of the invention.

Figure 7 shows a sample of seating chart in one embodiment of the invention.

Figure 8 is a code diagram of a code segment which implements the tentative reservation aspect of one embodiment of the invention.

DETAILED DESCRIPTION

Referring to **Figure 1**, it will be seen that the operator of a venue implements the disclosed reservation system to allow remotely located users to reserve specific seating for specific events 1. By doing so, the operator initiates those certain actions necessary to displaying an internet web site to all online users 2. A prospective customer for the venues offering(s) logs onto the internet 3 and acquires the aforesaid internet web site 4 which implements a reservation system for the venue. The prospective customer can be connected to the internet by any conventional means, yet this by no means implies that the wide-area network must be what is commonly referred to as the "internet." Upon contact by the prospective customer, an inquiry is directed to the appropriate database, which may be located concurrent with the primary server hosting the reservation system software program or may be located remotely, such as at the physical location of the venue, asking for a return of information to the prospective customer of all appropriate information contained therein relative to his inquiry 5. The prospective customer indicates a desired seat or seats through conventional computer input means and directs that information back to the server hosting the code necessary to the implementation of the reservation system 6. Upon contact 7, the server again makes an appropriate database query and returns to the prospective customer all pertinent information relating to that selection, such as which seats are still available for the chosen performance, airline flight, boxing match, etc. The prospective customer is then presented with a representation of all then available seating for the selected venue 8. From this representation, the prospective customer makes a selection of a seat or seats by indicating such through a mouse click, keyboard entry or other means, such as but not limited to a touch screen. Selection through voice command via a speech interface is also within the scope and contemplation of the invention. Simultaneously, the server, through the coding necessary to implement the reservation system, creates a temporary customer identification 9 that is used to associate this potential customer with the customer's later selections and permit system use by multiple simultaneous users. Once the customer has made the seat selection, the system requests payment information from the customer 10. That information is processed through conventional internet or other electronic means, and once the information and payment are verified 11a, the customer information as supplied in 10 is made permanent, and the seat or seats selected are removed from inventory and blocked from duplicate sale, both graphically when presented to the next prospective customer and in the database where information for accounting and administrative purposes is retained. If the customer's payment information cannot be verified 11b, then the customer is given an opportunity to correct the information or start over with a new transaction. Upon verification of the customer's payment information, the customer receives a confirmation of the transaction 13 containing all appropriate reference information for the customer's records.

Referring to **Figure 2**, it will be seen that, for example, a user in Houston 13 is planning to vacation in New York and wishes to see a play at a dinner theater there that utilizes the system for ticketing and reservations 15. The user, in Houston, or in any other location worldwide, connects to the internet in the conventional way and retrieves the appropriate web site through a graphical browser from a host server located in, say, Anaheim, California 14. Notably, the client node is unafilliated with the host server, i.e., need not have dedicated hardware or software to access the system. Rather, the system may be made accessible through any conventional web browser. The host server serves a seating chart for the dinner theater to the user such that the user is able to see the exact seating arrangement of the remote dinner theater and select the exact seat or seats desired for the performance of choice. Such additional information as is appropriate can be provided to the remote user to assist in making an informed decision as to which seat or seats are desired. By way of example, the host server may serve a view as seen from a particular seat tentatively selected. For example, in one embodiment, the tentative selection may be through clicking on the seat or merely detection of the cursor over the seat without a click. While in this example, the "venue" is a dinner theater, venue as used herein is not so limited and is deemed to include any forum for which seating space is sought, including without limitation, airplane or other transportation, stadiums, theaters, and the like.

Referring to **Figure 3**, it will be seen that in view (a) of the user selected venue all seats at table P11 17 and at table S14 18 have been previously taken and are so indicated by the graphical representation of an "X" over those seats. The potential customer wants to seat a party of four at table S1 16 and so indicates by clicking the mouse on those four seats or by so indicating through alternative standard computer input means. Once his payment method is verified, the selected seats are removed from inventory and so indicated on the graphical representation by placing an "X" over those seats 19 while retaining the "X" over those seats previously sold at table P11 20 and table S14 21. The next prospective customer is advised that these seats are no longer available for this performance by the new graphical representation that is the subsequent customer's first viewing screen upon entry into the system. In the event that two prospective customers wish to reserve the exact same seat or seats, that prospective customer who first receives validation of the payment method is given those seats, while the other prospective customer is notified that the seats desired have already been sold and offers a chance to select other seating.

Figure 4 shows the screens presented to a user when accessing the system and progressing through the process of selecting a specific seat or seats, and reserving and paying for them. Figure 4(a) is where the first screen presented shows links to available performances for that selected venue 22. Figure 4(b) is the second screen 23 and shows a view of the seating available for that venue with seats that have already been taken crossed off with an "X" 24. The hypothetical user wants to have a party of four sit at table S11 25 and selects the four seats around that table by clicking on them with the mouse. As the mouse moves the cursor over individual seats, the seat number appears in the window at the bottom of the user's screen 26 and when the user clicks on a seat, it is added to a running tally of the seats the user has already taken 27. Only seats having not previously been taken show up in the window 26 when the cursor passes over them. After completing the selections, the user clicks on the "Reserve Seats" button and Figure 4(c) shows the next screen which requests payment information 28. The user enters the required information and again clicks the "Reserve Seats" button 29. Figure 4(d) is the next screen and it tells the user that the payment method has been accepted (or rejected) and relates information about the transaction 30 such as a transaction code and a receipt number that can be used as a ticket or as a voucher with which to redeem a ticket or tickets at the venue box office upon arrival for the performance. Finally, Figure 4(d) shows the opening screen the next visitor to the system is presented with, which is the same set of screens except that the seats reserved by the hypothetical user 31 are marked off as already taken.

Figure 5 is a code diagram of one embodiment of the invention. The ticketing and reservation system of the present invention, in one particular embodiment thereof, includes a computer program operating on a server for a wide area network (WAN), generally described by the flow chart of Figure 1 and the accompanying code example which implements one embodiment of the instant invention. First, when a user accesses the system means is provided to initialize the process and return to the user a menu from which he selects his venue of interest. This can be a selectable menu arranged by artist or date or time or specific theater or football team or baseball team or name or activity or any combination thereof such that the user is given sufficient information from which to make a decision. An example would be someone looking for the next event at a given theater at a time that starts at 7:00pm. One of many possible series of computer instructions to perform this function is:

```
< -.Send database query to retrieve all venues that are currently
available in the system - >
< - Server receives and processes query - >
< - Query is looped until all available performances and venues are
retrieved. - >
< - Markup Language engine converts result to display compatible
format for output to client computer - >
< - Begin normal markup language here - >
< - Begin reservation process selecting the event date/time next to
the desired venue ->
```

Then, upon user submittal, the server initializes the process of returning to the user all available seats:

```
< -.Send database query to retrieve all seats that are currently
available in the system for this event - >
< - Server receives and processes query - >
< - Query is looped until all available seats are retrieved. - >
```

< - markup language engine converts result to markup language format for output to client computer - >

< - Begin normal markup language here - >

< - Continue reservation process by selecting the desire seat or seats ->

Then, upon user submittal, a new customer record is created in the Wide Area Network server and the system is notified which database to connect to to fulfill the users request(s):

< - Send database command to insert new record in customer database and obtain record id - >

< - Send database command to insert new record in reservation "order" database and obtain record id - >

< - Send database command to insert new record for each selected seat in the reservation "detail" database - >

< - Begin normal markup language here - >

< - Continue reservation process by requesting client payment information - >

Then, upon user submittal the information is passed for verification:

< - Submit client information for verification - >

< - if verification is successful, send database command to update customer record in customer database with information previously collected - >

< - if verification is successful, send database command to update reservation record in reservation "order" database with verification information - >

< - if verification is successful, send database command to remove selected seats from seat inventory database and marked as no longer available for future selection - >

< - Markup language engine converts result to markup language format for output to client computer - >

< - Begin normal markup language here - >

< - If verification is successful, confirmation is generated via Markup language engine to markup language format for output to client computer - >

```
        < - if verification is unsuccessful, a failure notice is generated via
        Markup language engine to markup language format for output to
        client computer - >
        < - if verification is unsuccessful, client is presented with option to
        provide his payment information again or abandon his reservation
        - >
```

While this is one preferred form of the code there are many other code sequences that will perform the same function that will be immediately obvious to one skilled in the art.

Figure 6 is a flow diagram of a host server operation in response to a venue request in an alternative embodiment of the invention. A host server receives a venue request at functional block 600. Then, at functional block 602, the host server serves a seating chart for that venue to the requesting client node. In one embodiment, the seating chart is served as a HTML page with active links for the seats which are currently available. In another embodiment, the seating chart may be served as a Java applet. At functional block 604, the host server receives a tentative seat selection from the client node. Then at decision block 606, a determination is made on the host server whether the tentative seat selection selected at functional block 604 would orphan one or more seats. An orphan seat is deemed to be any single seat that has no available a? seat within the row.

Some venues may wish to force alternative seat selections if the tentative seat selections would orphan seats. Following alternative seat selection and the criterion for doing so may differ from venue to venue, as well as depending on the existing availablity of comparable seating that would not orphan seats. A determination is made by the host server seat at decision block 608 whether the selected venue wishes to force alternative seat selection. If the venue does not wish to force alternative seat selection, the host server may optionally nevertheless send a request that alternative seats be selected at functional block 610. A determination is made at decision block 612 after the request that alternative seats be selected is sent at functional block 610 whether the client node has accepted the request. Alternatively, if the venue wishes to force alternative seat selection at functional block 608, a message indicating that the selected seats are unavailable and possibly explaining that the venue does not permit seat selection that would orphan seats sent to the client node at functional block 614.

If at decision block 612, the request has been accepted, suggested alternative seat selections may be sent at functional block 616. The alternative suggested seat selections may also be sent subsequent to the notice that selected seats are unavailable at functional block 614. After the suggested alternative seat selection is sent at functional block 616, the determination is made whether the suggestion has been accepted by the client at decision block 618. If the suggested alternative seat selection is not accepted, a determination is made at decision block 620 whether an alternative seat selection has been received, e.g., different than the original tentative seat selection and different from the suggested alternative seat selection. If not, the interaction with the client ends. If it has, the server makes an orphan determination as to the new alternative seats. After the request is not accepted at decision block 614, or no seats would be orphaned at decision block 606, the server places a tentative reservation on the seats at functional block 622. Alternatively, the tentative reservation may be made prior to the orphan determination and cancelled if alternative seats are forced or selected. Once the tentative reservation is placed on the seats, all subsequent users accessing the server will be served the seating chart with the tentative seats shown in a representation that is different from both the representation of sold seats and the representation of unsold seats. This is termed the tentative reserve representation. In one embodiment, this takes the form of a question mark appearing over the seats, as displayed on the seating chart.

At functional block 624, a request for payment information is issued by the host server. A determination is made at decision block 626 if the payment information received is valid. If the payment information is not valid, a request is made by the server at functional block 628 for alternative payment information coupled with an indication that the prior payment information was unacceptable. If the alternative payment information is received at decision block 630, that payment information is then verified at functional block 626. If not, the tentative reservation is removed at functional block 632. This returns the representation of the seat on subsequently served seating charts to the unsold representation. If the payment information is valid at decision block 626, the reservation is made permanent, and the representation of the seat on the seating chart is changed to the permanently reserved reservation representation. A printable receipt, ticket, or voucher sufficient to permit access to the event is then served to the client node.

Figure 7 shows a sample of seating chart in one embodiment of the invention. As shown in the figure, seats C and D in row 10 are shown in a tentatively reserved representation. The "Xs" mark seats which are permanently reserved, and the circles with no indication represent seats in the unsold state. Seat 10(i) would be considered an orphan seat.

Figure 8 is a code diagram of a code segment which implements the tentative reservation aspect of one embodiment of the invention. Numerous other codings are, of course, possible.

Those having skill in the art to which the present invention pertains will now understand that there are virtually unlimited uses for the present invention. By way of example, the present invention may be readily used to reserve specific seats on commercial airliners or reserve specific staterooms on a cruise ship, as well as for reserving seats for any venue from community theater or little league baseball to major league sporting events.

In the foregoing specification, the invention has been described with reference to specific embodiments thereof. It will, however, be evident that various modifications and changes can be made thereto without departing from the broader spirit and scope of the invention as set forth in the appended claims. The specification and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense. Therefore, the scope of the invention should be limited only by the appended claims.

CLAIMS

What is claimed is:

1. A method comprising:
transmitting specific seat availability information to a user across a wide area network to an unaffiliated client node;
receiving an indication of tentative seat selection from the user across the wide area network; and
accepting payment from the user; and
denoting the seat as unavailable to future users.

2. The method of claim 1 further comprising:
indicating to any subsequently requesting client node that the tentative seat selections are tentatively unavailable until a transaction is completed or voided.

3. The method of claim 1 further comprising:
identifying if the tentative seat selection would orphan at least one unselected seat.

4. The method of claim 3 further comprising:
denying a reservation of the tentative seat selection if the tentative seat selection would orphan at least one unselected seat.

5. The method of claim 3 further comprising:
requesting that the user make a different selection if the tentative selection would orphan at least one seat; and
suggesting a different selection that would not orphan at least one seat.

6. The claim of 5 further comprising:
denying a reservation of the tentative seat selection if the tentative seat selection would orphan at least one selected seat if alternative seats of a same class are available that would not orphan a seat.

7. The method of claim 2 wherein indicating comprises:
changing a representation of a seat from a first representation to a second representation; and
transmitting the second representation to a client node.

8. The method of claim 7 further comprising:
changing the representation of the seat from the second representation to a third representation if the transaction is completed; and
changing the representation of the seat from the second representation to the first representation of the transaction is voided.

9. The method of claim 1 further comprising:
providing additional information about the tentative seat selection to the user across the wide area network.

10. A system comprising:
a host server coupled to a wide area network (WAN); and
a database coupled to the server, the database, the database populated with entries reflecting seating availability for at least one venue;
wherein the host server serves seating information on demand responsive to a request of an unaffiliated node.

11. The system of claim 10 wherein seating information comprises:
a seating chart for the venue showing available seats in a first representation, tentatively unavailable seats in a second representation, and unavailable seats in a third representation;

12. The system of Claim 10 wherein the server serves pages providing a graphical user interface from which a user may select specifically desired seats from the venue.

13. A method comprising:
requesting a venue at an unafilliated client node from a host server;
receiving a seating chart for the venue;
tentatively selecting seats shown in a first representation from the seating chart; and providing payment information.

14. The method of claim 13 further comprising;

receiving the seating chart showing the tentatively selected seats in a second representation while a transaction is pending.

15. The method of claim 14 further comprising;

receiving the seating chart showing the tentatively selected seats in a third representation once the transaction is complete; and
receiving a printable electronic receipt sufficient to permit access to the venue.

16. A method comprising:

(a) communicating on demand, from an information server through a wide area network to a device connected to the wide area network information from a database populated by a multiplicity of entries denoting availabilityfor a venue;

(b) displaying the information such that an end-user connected to the wide area network can view the information on a client node unaffiliated with the server as an aid in determining a best then available space conforming to a need of the end-user;

(c) providing over the wide area network to the end user the capability of interactively selecting a time, a space, and a seat of choice;

(d) accepting over the wide area network from the end user a payment for one of the time, the space, and the seat selection of choice;

(e) returning over the wide area network to the end user verification of the successful completion of the payment.

17. The method described in claim 16 wherein the space, the time, and seat sought is for a theater or theater type setting.

18. The method described in claim 16 wherein the space, the time, and the seat sought is for a stadium type setting.

19. The method described in claim 16 wherein the space or time or seat or seating sought is for an airplane or airliner.

20. The method described in claim 16 wherein the space or time or reservation sought is accommodations on a cruise ship.

21. The method described in claim 16 wherein a communication connection between the information server and the end user includes one of a wire, a cable, and a telephone line connection.

22. The method described in claim 16 wherein a communication connection between the information server and the end user includes a satellite link.

23. The method described in claim 16 wherein a communication connection between the information server and the end user includes a wireless link.

24. The method described in claim 17 wherein the communication connection between the information server and the end user is wireless.

25. A method comprising:

receiving at a server a request for a venue from at least one client node remote from and unaffiliated with the server;

transmitting from the server an indication of specific availability responsive to the request, the indication of specific availability directed to the client node;

receiving a specific indication of a client preference at the server;

removing the client preference from any future indication of specific availability.

26. The method of claim 25, wherein the specific availability includes seating, further comprising:

retrieving from a database an image showing a view from a seat indicated by the client preference;

transmitting the image to the client.

27. The method of claim 25, wherein the indication of specific availability includes a graphical representation of at least a portion of a seating chart for the venue, and wherein the graphical representation shows availability seats in a first representation and previously sold seats in a second representation.

28. The method of claim 25, wherein the indication of the specific availability is transmitted as one of a hypertext markup language page and a java applet.

29. The method of claim 27 further comprising:

linking the representation of a seat to an image of a view from that seat.

30. The method of claim 25 further comprising:

accepting payment information at the server sufficient to permit access to the specific client preference;

conducting an electronic payment transaction; and

providing an electronic receipt.

31. A method comprising:

requesting information about a venue across a wide area network (WAN) from a client node to be supplied by an unaffiliated server node;

receiving an indication of specific availability at the client node;

selecting from a plurality of specific availability options a specific client preference; and

receiving an indication that the specific client preference has been reserved through the server node.

32. The method of claim 31 wherein the indication of specific availability includes a graphical representation of at least a portion of a seating chart for the venue, and wherein the graphical representation shows available seats in a first representation and previously sold seats in a second representation.

33. The method of claim 32 wherein selecting comprises:

clicking on a desired seat.

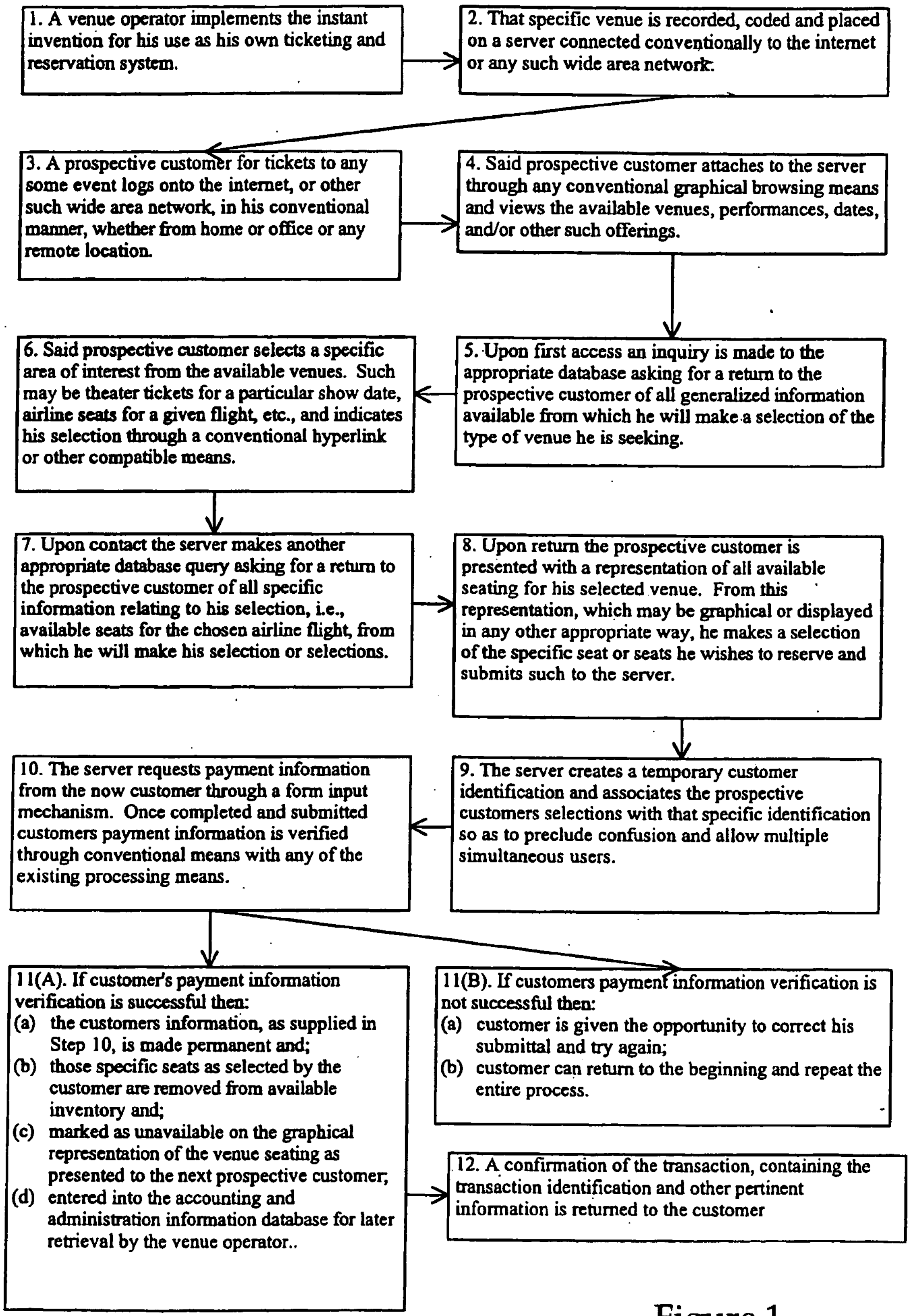
34. The method of claim 33 further comprising:

receiving an image of a view from the desired seat responsive to the clicking.

35. The method of claim 31 further comprising:

supplying payment information for the specific client preference; and

receiving an electronic receipt sufficient to permit access to the specific client preference.



1. A venue operator implements the instant invention for his use as his own ticketing and reservation system.

2. That specific venue is recorded, coded and placed on a server connected conventionally to the internet or any such wide area network.

3. A prospective customer for tickets to any some event logs onto the internet, or other such wide area network, in his conventional manner, whether from home or office or any remote location.

4. Said prospective customer attaches to the server through any conventional graphical browsing means and views the available venues, performances, dates, and/or other such offerings.

5. Upon first access an inquiry is made to the appropriate database asking for a return to the prospective customer of all generalized information available from which he will make a selection of the type of venue he is seeking.

6. Said prospective customer selects a specific area of interest from the available venues. Such may be theater tickets for a particular show date, airline seats for a given flight, etc., and indicates his selection through a conventional hyperlink or other compatible means.

7. Upon contact the server makes another appropriate database query asking for a return to the prospective customer of all specific information relating to his selection, i.e., available seats for the chosen airline flight, from which he will make his selection or selections.

8. Upon return the prospective customer is presented with a representation of all available seating for his selected venue. From this representation, which may be graphical or displayed in any other appropriate way, he makes a selection of the specific seat or seats he wishes to reserve and submits such to the server.

9. The server creates a temporary customer identification and associates the prospective customers selections with that specific identification so as to preclude confusion and allow multiple simultaneous users.

10. The server requests payment information from the now customer through a form input mechanism. Once completed and submitted customers payment information is verified through conventional means with any of the existing processing means.

11(A). If customer's payment information verification is successful then:
(a) the customers information, as supplied in Step 10, is made permanent and;
(b) those specific seats as selected by the customer are removed from available inventory and;
(c) marked as unavailable on the graphical representation of the venue seating as presented to the next prospective customer;
(d) entered into the accounting and administration information database for later retrieval by the venue operator..

11(B). If customers payment information verification is not successful then:
(a) customer is given the opportunity to correct his submittal and try again;
(b) customer can return to the beginning and repeat the entire process.

12. A confirmation of the transaction, containing the transaction identification and other pertinent information is returned to the customer

Figure 1

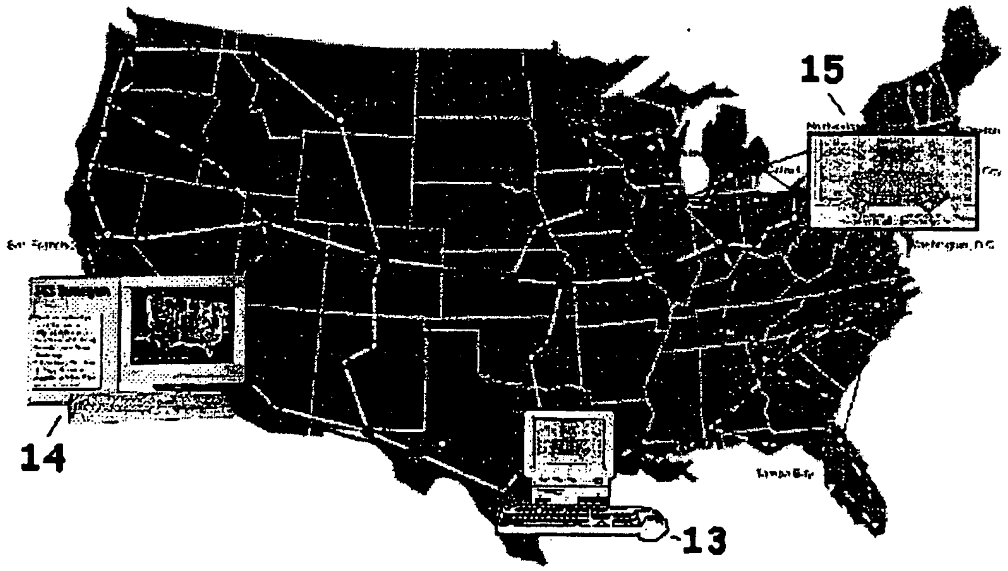


Figure 2

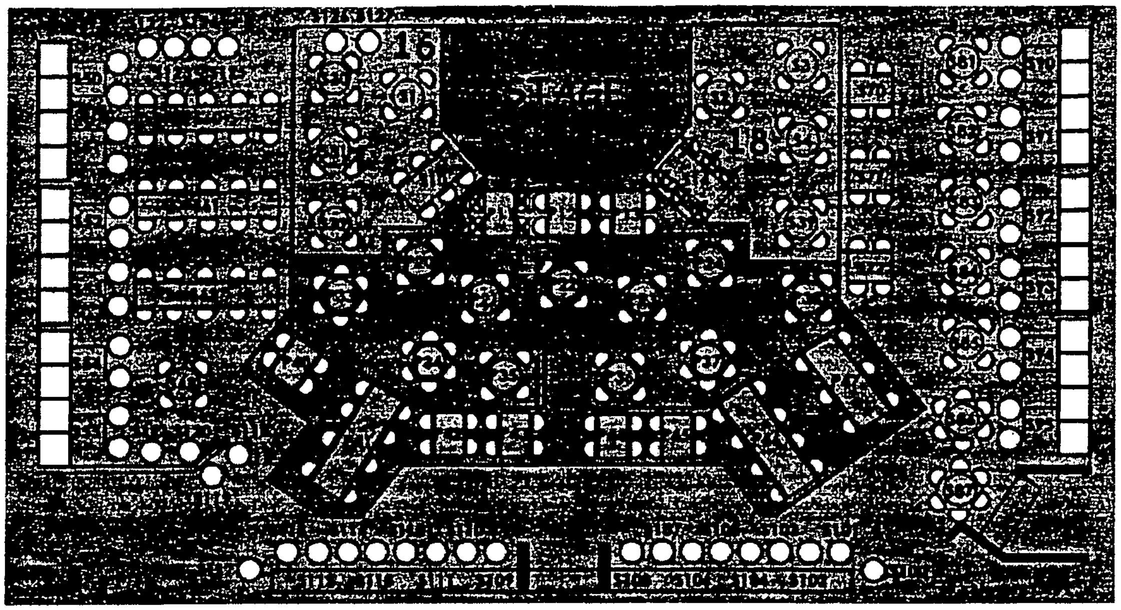


Figure 3a

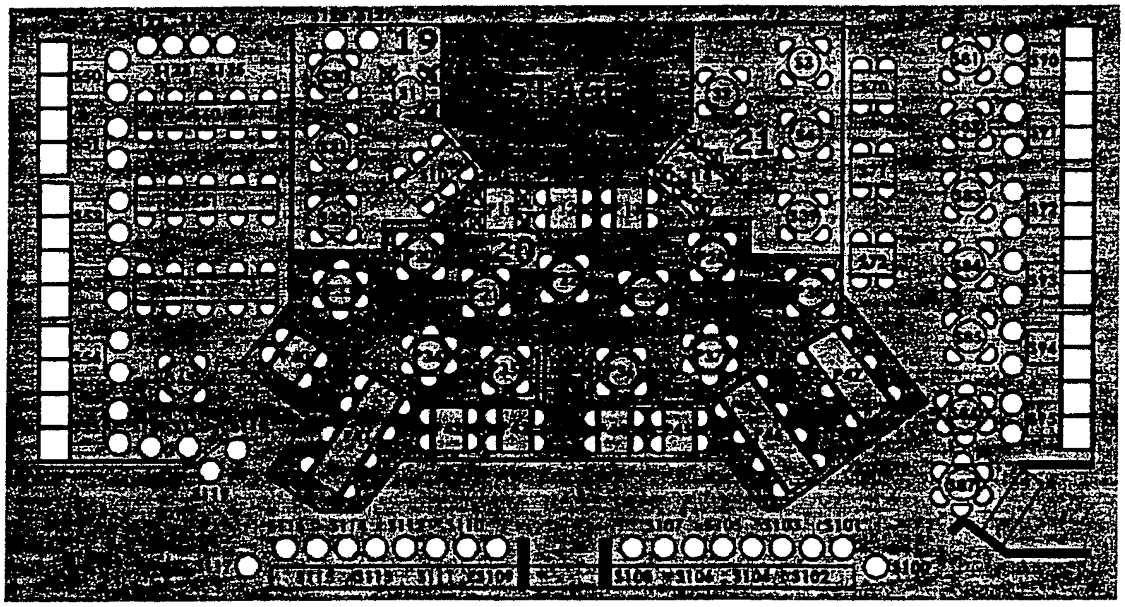


Figure 3b

Start Figure 4

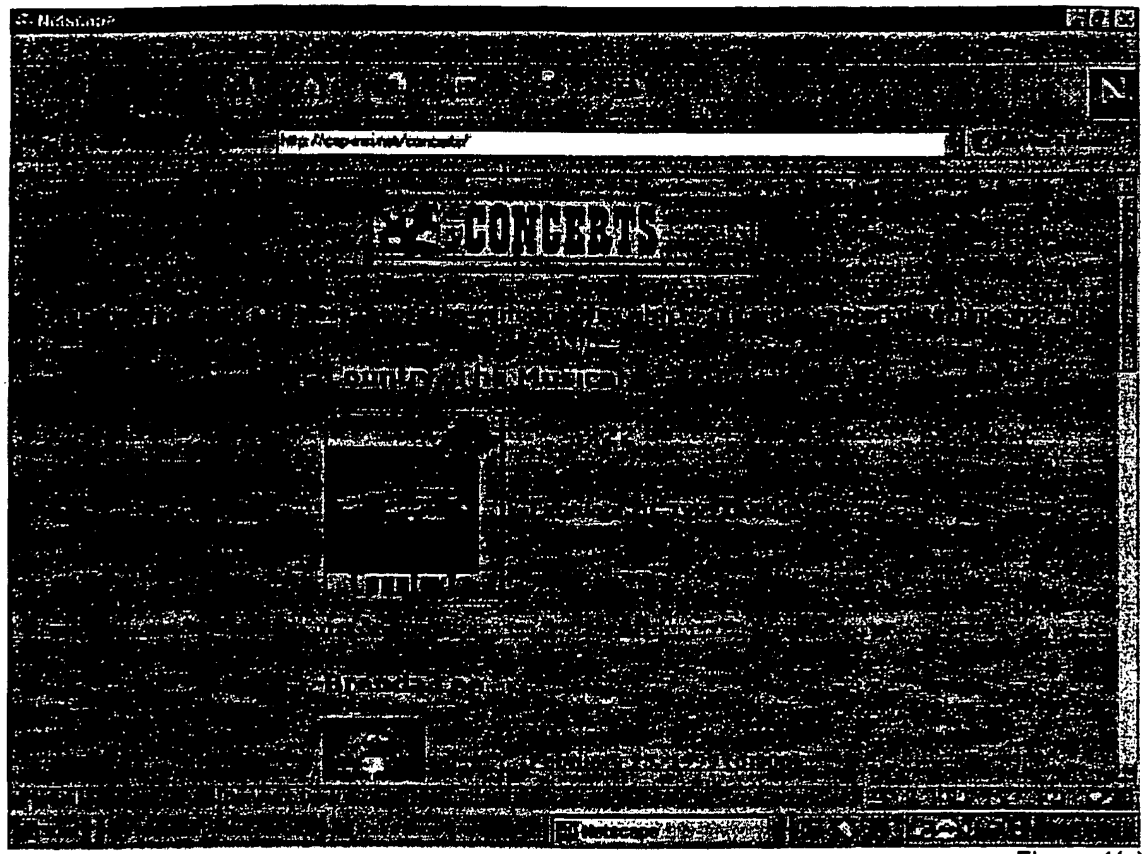


Figure 4(a)

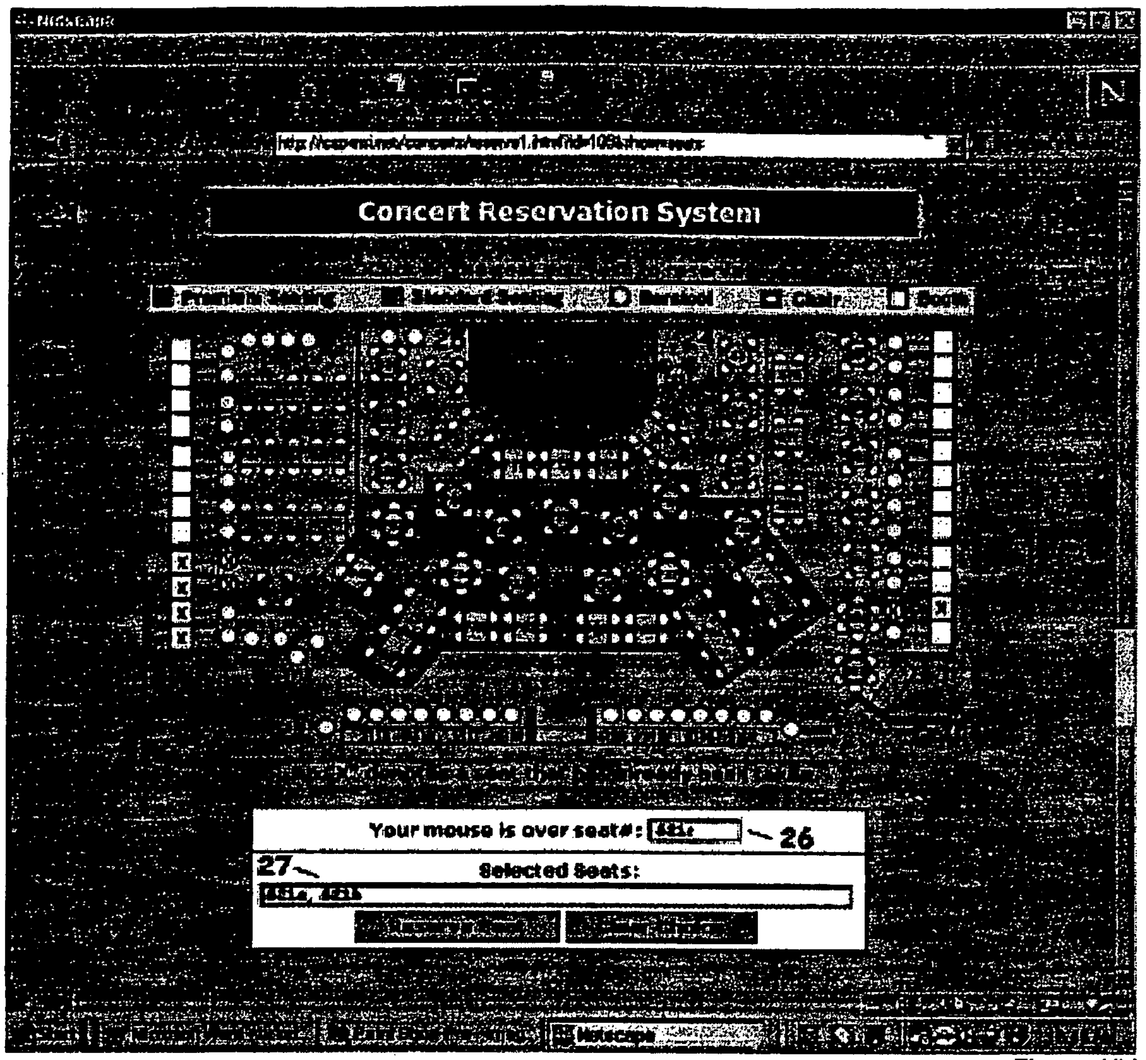
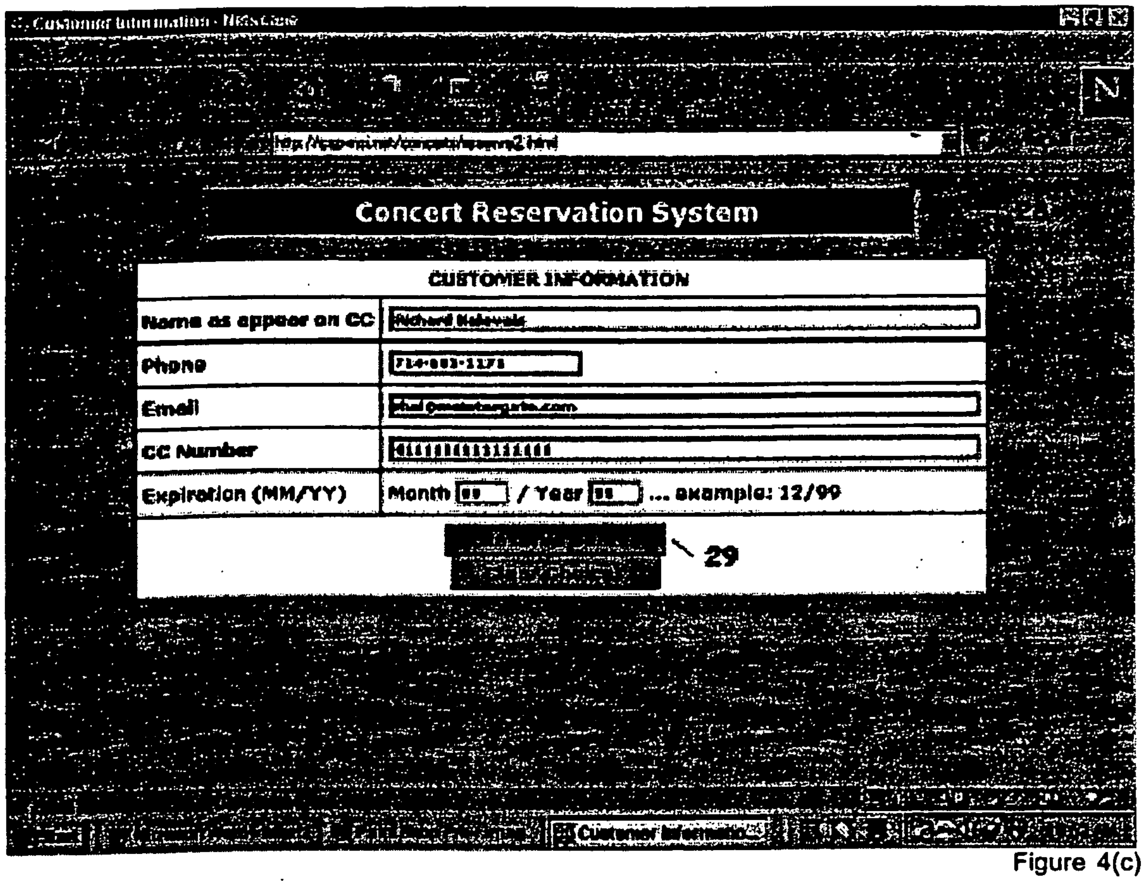

Concert Reservation System
Your mouse is over seat#:
26
27
Selected Seats:

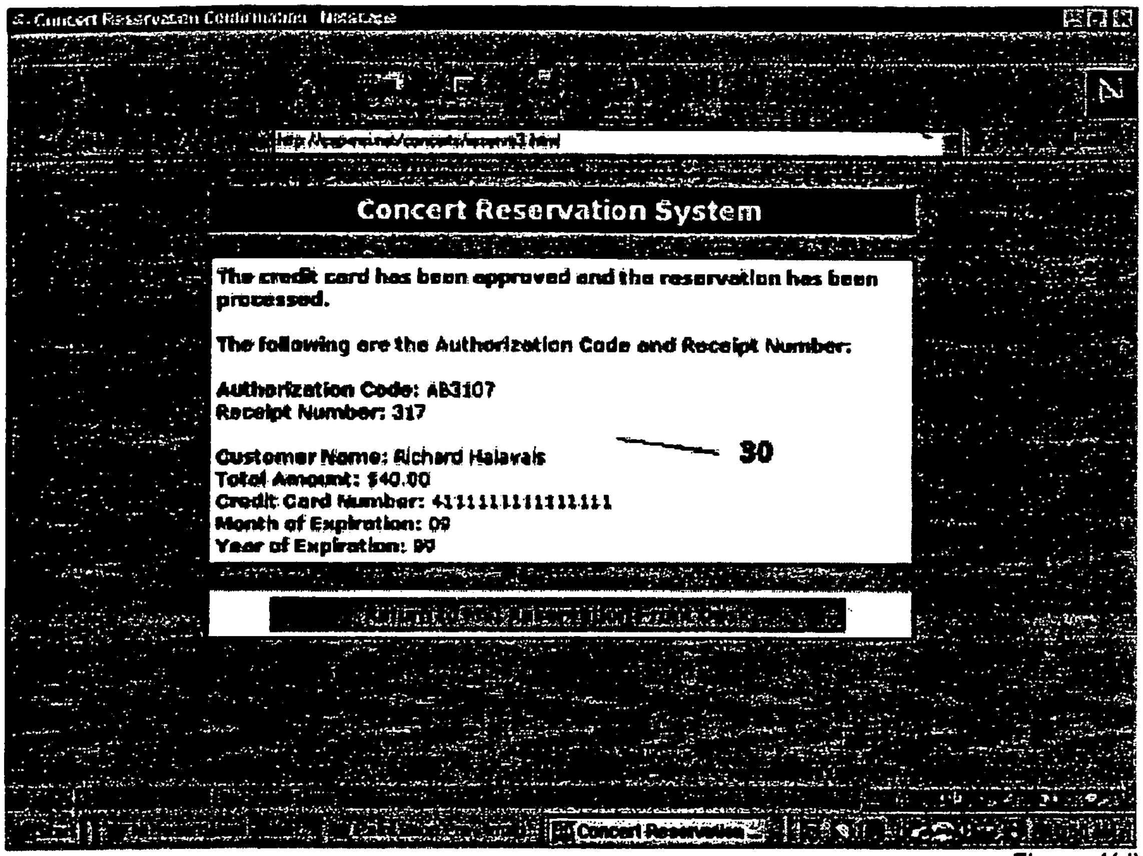
Figure 4(b)



Concert Reservation System

CUSTOMER INFORMATION	
Name as appear on CC	Richard [illegible]
Phone	714-883-1171
Email	[illegible]
CC Number	4111111111111111
Expiration (MM/YY)	Month [illegible] / Year [illegible] ... example: 12/99

Figure 4(c)


Concert Reservation Confirmation - Netscape
Concert Reservation System
30
Concert Reservation

The credit card has been approved and the reservation has been processed.

The following are the Authorization Code and Receipt Number:

Authorization Code: AB3107
Receipt Number: 317

Customer Name: Richard Halavais
Total Amount: $40.00
Credit Card Number: 4111111111111111
Month of Expiration: 09
Year of Expiration: 99

Figure 4(d)

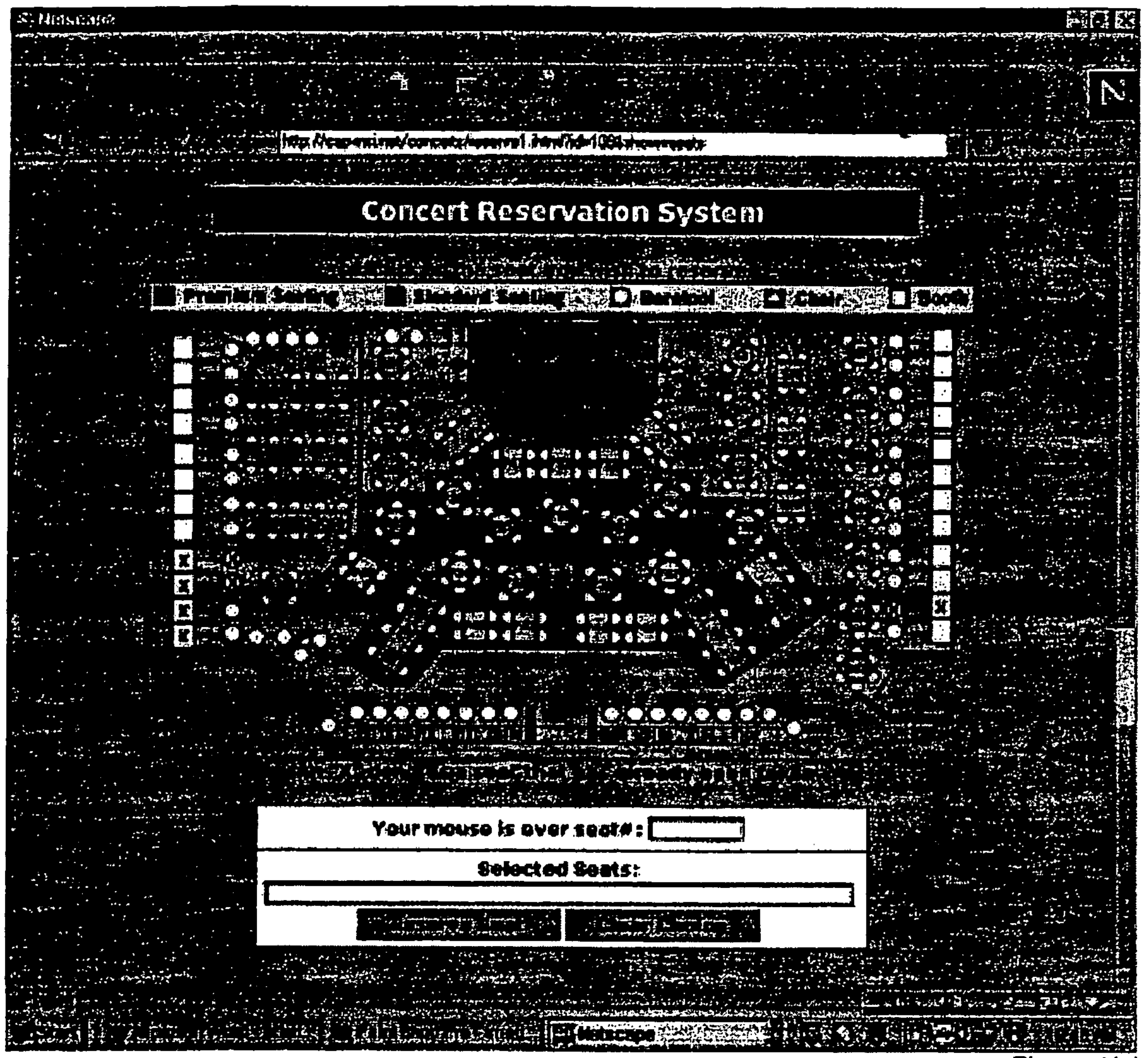


Figure 4(e)

End Figure 4

Start Figure 5

```
Begin Program
<Markup language>
<REM --- Imports the file "datasource.inc" which creates variable "datasource"
which is used to tell markup language datasource to connect to.
-- >
<INCLUDE NAME="database\datasource.inc">
<REM --- In case a database or other type of error occurs, this display the error
message.  --- >
<ERROR>
<FONT FACE="Verdana, Arial" SIZE="+1"><B>An Error Has
Occurred</B></FONT><P>
<FONT FACE="Verdana, Arial" SIZE="-1"><B>Error Message = :i_errortext
</B></FONT><P>
<FONT FACE="Verdana, Arial" SIZE="-1"><B>database Error =
:i_databaseerrortext</B></FONT><P>
<FONT FACE="Verdana, Arial" SIZE="-1"><B>database Error =
:i_databaseerrorstmt
</B></FONT><P>
</ERROR>
<REM --- Begin normal markup language here                    --- >
<markup language>
<HEAD>
```

```
<TITLE>Ticketing &amp; Reservation System</TITLE>
</HEAD>
<BODY BACKGROUND="images/background.jpg" TEXT="#000000"
LINK="#006666" VLINK="#006666">
<CENTER>
<IMG SRC="images/masthead_concerts.gif" HEIGHT=60 WIDTH=280><P>
<FONT FACE='Verdana,Arial' SIZE=-1><B>To begin reserving your seat(s),
please select the concert date/time you wish to attend next to the performer you
want to see:</B></FONT>
<TABLE BORDER=0 CELLSPACING=5 CELLPADDING=5>
<REM --- Begin database query to retrieve all performances that is         --- >
<REM --- currently available.  Will loop until all available          --- >
<REM --- performing artists and their performances are listed.       --- >
<REM --- Part of the return from the query are the links that will  --- >
<REM --- take you to the next step of the reservation.               --- >
<database DBNAME=":datasource"
        database="SELECT id, name, picture, sequence
        FROM category
        WHERE active=1 AND parent=-1
        ORDER BY sequence"
        ALIAS="concert">
        <databaseFETCH ALIAS="concert">
        <WHILE NOTALIAS=i_databaseempty>
        <TR>
        <TD COLSPAN=2>
        <FONT FACE="Verdana,Arial"
SIZE=+1><B>:concert_name</B></FONT>
        </TD></TR><TR><TD VALIGN="top">
        <img src=images/:concert_picture align=top border="1">
        </TD><TD>
        <Markup language DBNAME=":datasource"
        database="SELECT id, name, date, time
        FROM category
        WHERE active=1 AND parent=:concert_id
        ORDER BY date, time"
        OUTPUT="<FONT FACE='Verdana,Arial' SIZE=-1><B>
        <A HREF='reserve1.ihtml?&id=:1'>:2</A>
        </B></FONT><p>">
        </TD></TR><TR><TD COLSPAN=2><HR></TD></TR>
        <databaseFETCH ALIAS="concert">
        </WHILE ALIAS=i_databaseempty>
</database ALIAS="concert">
</TABLE>
</BODY>
</markup language>
```

processing then passes to:

```
<Markup language>
<REM --- Imports the file "datasource.inc" which creates the        --- >
```

IV

```
<REM --- variable "datasource" which is used to tell Markup language which      -

-- >

<REM --- database datasource to connect to.                    --- >

<INCLUDE NAME="database\datasource.inc">

<REM In case a database or other type of error occurs, this will display the error

message. >

<ERROR>

<FONT FACE="Verdana,Arial" SIZE="+1"><B>An Error Has

Occurred</B></FONT><P>

<FONT FACE="Verdana,Arial" SIZE="-1"><B>Error Message =

:i_errortext</B></FONT><P>

<FONT FACE="Verdana,Arial" SIZE="-1"><B>database Error =

:i_databaseerrortext</B></FONT><P>

<FONT FACE="Verdana,Arial" SIZE="-1"><B>database Error =

:i_databaseerrorstmt</B></FONT><P>

</ERROR>

<REM --- Begin normal markup language here --- >

<markup language>

<HEAD>

<TITLE>Ticketing &amp; Reservation System - Select Seat(s)</TITLE>

</HEAD>

<REM --- All seats clicked will pass its information to a input box, "newseats".

When done, the information will be passed to "process.ihtml" and be processed

by "reserve2.ihtml". --- >
```

V

```
<REM --- This code allows the ability to select multiple seats before proceeding to
the next step of reservation process.  Other features include listing the selected
seats in the "number" text box to show the seats that has been clicked.  This script
also alters the button text to be grammatically correct. --- >
<SCRIPT LANGUAGE="JavaScript">
<!--
function selectseat(idnum, seatnum)
{
reserved = document.seats.number.value;
if (reserved == "")
{
document.seats.newseats.value = "<Markup language DBNAME=:"+"datasource
database='INSERT INTO basket (custid, pid, qty) VALUES (:"+"custid,
"+idnum+", 1)'><database DBNAME=:"+"datasource database='SELECT cost
FROM products WHERE id="+idnum+"'><databaseFETCH><iEQ
NAME='ticketprice' VALUE=:"+"1></database><iEQ NAME='total'
VALUE=<iEVAL EXPR=':"+"ticketprice + :"+"total' PREC='2'>><Markup
language DBNAME=:"+"datasource database='INSERT INTO orderdetail (pid,
oid, qty, sell) VALUES ("+idnum+", :"+"oid, 1, :"+"ticketprice)'>";
document.seats.number.value = seatnum;
document.seats.button1.value = "Reserve Seat";
document.seats.button2.value = "Clear Choice";
}
else
{
document.seats.newseats.value += "<Markup language
DBNAME=:"+"datasource database='INSERT INTO basket (custid, pid, qty)
VALUES (:"+"custid, "+idnum+", 1)'><database DBNAME=:"+"datasource
database='SELECT cost FROM products WHERE
id="+idnum+"'><databaseFETCH><iEQ NAME='ticketprice'
VALUE=:"+"1></database><iEQ NAME='total' VALUE=<iEVAL
EXPR=':"+"ticketprice + :"+"total' PREC='2'>><Markup language
DBNAME=:"+"datasource database='INSERT INTO orderdetail (pid, oid, qty,
sell) VALUES ("+idnum+", :"+"oid, 1, :"+"ticketprice)'>";
document.seats.number.value = reserved+", "+seatnum;
document.seats.button1.value = "Reserve Seats";
document.seats.button2.value = "Clear Choices";
}
}
function clear()
{
document.seats.newseats.value = "";
}
//->
</SCRIPT>
<BODY BACKGROUND="images/background.jpg" TEXT="#000000"
LINK="#006666" VLINK="#006666">
```

```
<REM --- Begin database query to retrieve the last id number from table
"orderdetail" for the purpose of creating a new instance of the stage image
whenever a new order has occurred. This will prevent the browser from
displaying an old floor image from its cache as a different image name is called
every time the file is run. --- >
<database DBNAME=":datasource"
        database="SELECT max(id)
        FROM orderdetail">
        <databaseFETCH>
        <iEQ NAME="imagenum" VALUE=:1>
        <databaseFETCH>
</database>
<iEQ NAME="imagetype" VALUE=".jpg">
<CENTER>
<TABLE BGCOLOR='#000000' BORDER=0 CELLPADDING=5 CELLSPACING=0
WIDTH=500>
<TR><TD><CENTER>
<FONT FACE='Verdana,Arial' SIZE=+1 COLOR='#FFFFFF'><B>
Concert Reservation System
</B></FONT>
</CENTER>
</TD></TR></TABLE></CENTER><p>
<REM - Loads the core image that will be dynamically altered for use as the
image map. - >
```

VIII

```
<iIMAGEFROMFILE NAME="stage" FILENAME="images/stage.jpg"
TYPE="jpeg">
<MAP NAME="stage" BORDER=0>
<REM --- Begin database query to retrieve all seat information for the image map
"stage".  Will loop until all available seat information for this particular
performance is listed.     --- >
<REM --- The return query will be used only if the "active" attribute of that
particular seat is marked "true" or "1".  --- >
<REM --- If active, the seat's information, including image map coordinates will
be displayed.  Otherwise, the seat on the image map will be x'd out according to
the x and y coordinates associated with it.                    --- >
<database DBNAME=":datasource"
database="SELECT id, active, x, y, shape, corrds, name, mouseout, mouseover1,
mouseover2,  mouseover3
        FROM products
        WHERE catid=:id">
        <databaseFETCH>
        <iWHILE NOTALIAS=i_databaseempty>
        <iEQ NAME="active" VALUE=:2>
        <iCASE ALIAS="active" VALUE=1>
<AREA SHAPE=":5" COORDS=":6" HREF="javascript:selectseat(:1,':7')"
ALT="Seat # :7" OnMouseOut=:8 OnMouseOver=:9 :7 :10 :7 :11 >
        </iCASE ALIAS="active">
        <iCASE ALIAS="active" VALUE=0>
```

IX

```
        <iIMAGETEXT NAME="stage" TEXT="X" X=:3 Y=:4 COLOR="red">
        </iCASE ALIAS="active">
        <databaseFETCH>
        </iWHILE ALIAS=i_databaseempty>
</database>
</MAP>
<CENTER>
<FONT FACE="Verdana,Arial" SIZE="-1"><B>
Click on the seat you wish to reserve.
</B></FONT>
<TABLE BORDER=1 CELLPADDING=0 CELLSPACING=0>
<TR><TD>
<REM --- Converts the dynamically generated image, "stage", into a web friendly
image type - "jpeg".         --- >
<iIMAGEWRITE NAME="stage" FILENAME="images/ch_stage-:id-
:imagenum:imagetype" TYPE="jpeg" QUALITY="30">
<REM --- Deletes the temporary image "stage" as it is no longer needed. --- >
<iIMAGEDESTROY NAME="stage">
<IMG SRC="images/legend.jpg" BORDER=0 HEIGHT=20 WIDTH=584><BR>
<IMG SRC="images/ch_stage-:id-:imagenum:imagetype" BORDER=0
USEMAP="#stage" HEIGHT=315 WIDTH=584>
</CENTER>
</TD></TR></TABLE>
<FONT FACE='Verdana,Arial' SIZE=-1><B>
```

```
X
An "X" denotes a seat that has already been taken.
</B></FONT>
<FORM NAME="seats" ACTION="process.ihtml" METHOD="post">
<CENTER>
<TABLE BGCOLOR='#000000' BORDER=0 CELLPADDING=0 CELLSPACING=0>
<TR><TD>
<TABLE BGCOLOR='#FFFFFF' BORDER=0 CELLPADDING=5
CELLSPACING=1>
<TR><TD><CENTER>
<FONT FACE='Verdana,Arial' SIZE=-1><B>
Your mouse is over seat#: <INPUT TYPE="text" NAME="seatnum" SIZE=5>
</B></FONT></CENTER>
</TD></TR><TR><TD><CENTER>
<FONT FACE='Verdana,Arial' SIZE=-1><B>
Selected Seats:<BR><INPUT TYPE="text" NAME="number" SIZE=40>
<BR>
<INPUT TYPE="hidden" NAME="newseats" SIZE=50>
<INPUT TYPE="submit" NAME="button1" VALUE="Reserve
Seat">&nbsp;<INPUT TYPE="reset" NAME="button2" VALUE="Clear Choice"
onClick="clear()">
</B></FONT>
</CENTER>
</TD></TR></TABLE></TD></TR></TABLE></CENTER>
</FORM></CENTER></BODY>
```

```
</markup language>
```

which then passes to a template:

```
<Markup language>
<REM --- Imports the file "datasource.inc" which creates the variable
"datasource" which is used to tell Markup language which ODBC datasource to
connect to.
<iINCLUDE NAME="database\datasource.inc">
<REM --- In case a database or other type of error occurs, this will display the
error message. - >
<ERROR>
<FONT FACE="Verdana,Arial" SIZE="+1"><B>An Error Has
Occurred</B></FONT><P>
<FONT FACE="Verdana,Arial" SIZE="-1"><B>Error Message =
:i_errortext</B></FONT><P>
<FONT FACE="Verdana,Arial" SIZE="-1"><B>database Error =
:i_databaseerrortext</B></FONT><P>
<FONT FACE="Verdana,Arial" SIZE="-1"><B>database Error =
:i_databaseerrorstmt</B></FONT><P>
</ERROR>
<REM --- Copy a preformatted file, "empty.ihtml" into a new file,     --- >
<REM --- "reserve2.ihtml".                                             --- >
<iCOPYFILE SRC="empty.ihtml" DST="reserve2.ihtml">
```

XII

```
<REM --- Appends the information passed from "resevel.ihtml" to
"reserve2.ihtml" for process. Other html information is also passed.
--- >
<iFILE NAME="reserve2.ihtml" DATA=":newseats" OP="append">
<iFILE NAME="reserve2.ihtml" DATA="<INPUT TYPE='hidden'
NAME='total' VALUE=:total>" OP="append">
<iFILE NAME="reserve2.ihtml" DATA="</FORM>" OP="append">
<iFILE NAME="reserve2.ihtml" DATA="</CENTER>" OP="append">
<iFILE NAME="reserve2.ihtml" DATA="<P>" OP="append">
<iFILE NAME="reserve2.ihtml" DATA="</BODY>" OP="append">
<iFILE NAME="reserve2.ihtml" DATA="</markup language>" OP="append">
<REM --- Once the information is appended into "reserve2.ihtml", it will be
automatically push the page "reserve2.ihtml" to the browser.
--- >
<iREDIR URL="reserve2.ihtml">
```

which then combines information and passes to:

```
<Markup language>
<REM --- Imports the file "datasource.inc" which creates the variable
"datasource" which is used to tell Markup language which ODBC datasource to
connect to.            --- >
<iINCLUDE NAME="database\datasource.inc">
<REM --- In case a database or other type of error occurs, this will display the
error message. - >
<ERROR>
```

```
<FONT FACE="Verdana,Arial" SIZE="+1"><B>An Error Has
Occurred</B></FONT><P>
<FONT FACE="Verdana,Arial" SIZE="-1"><B>Error Message =
:i_errortext</B></FONT><P>
<FONT FACE="Verdana,Arial" SIZE="-1"><B>database Error =
:i_databaseerrortext</B></FONT><P>
<FONT FACE="Verdana,Arial" SIZE="-1"><B>database Error =
:i_databaseerrorstmt</B></FONT><P>
</ERROR>
<REM --- Copy a preformatted file, "empty.ihtml" into a new file,
"reserve2.ihtml". --- >
<iCOPYFILE SRC="empty.ihtml" DST="reserve2.ihtml">
<REM --- Appends the information passed from "reseve1.ihtml" to
"reserve2.ihtml" for process. Other html information is also passed. --- >
<iFILE NAME="reserve2.ihtml" DATA=":newseats" OP="append">
<iFILE NAME="reserve2.ihtml" DATA="<INPUT TYPE='hidden'
NAME='total' VALUE=:total>" OP="append">
<iFILE NAME="reserve2.ihtml" DATA="</FORM>" OP="append">
<iFILE NAME="reserve2.ihtml" DATA="</CENTER>" OP="append">
<iFILE NAME="reserve2.ihtml" DATA="<P>" OP="append">
<iFILE NAME="reserve2.ihtml" DATA="</BODY>" OP="append">
<iFILE NAME="reserve2.ihtml" DATA="</markup language>" OP="append">
<REM --- Once the information is appended into "reserve2.ihtml", it will be
automatically push the page "reserve2.ihtml" to the browser.
```

XIV

```
<iREDIR URL="reserve2.ihtml">
then (takes all information from index & reserve one and conbines it to with the
information in empty) then passes to:
<!Markup language>
<REM --- Imports the file "datasource.inc" which creates the variable
"datasource" which is used to tell Markup language which ODBC datasource to
connect to. --- >
<iINCLUDE NAME="database\datasource.inc">
<REM -- In case a database or other type of error occurs, this will display the error
message. -- >
<ERROR>
<FONT FACE="Verdana,Arial" SIZE="+1"><B>An Error Has
Occurred</B></FONT><P>
<FONT FACE="Verdana,Arial" SIZE="-1"><B>Error Message =
:i_errortext</B></FONT><P>
<FONT FACE="Verdana,Arial" SIZE="-1"><B>database Error =
:i_databaseerrortext</B></FONT><P>
<FONT FACE="Verdana,Arial" SIZE="-1"><B>database Error =
:i_databaseerrorstmt</B></FONT><P>
</ERROR>
<REM --- Begin normal markup language here                    --- >
<markup language>
<HEAD>
<TITLE>Ticketing &amp; Reservation System - Customer Information</TITLE>
```

```
</HEAD>
<BODY BACKGROUND="images/background.jpg" TEXT="#000000"
LINK="#006666" VLINK="#006666">
<REM ----- Get Customer Info ----- >
<CENTER>
<TABLE BGCOLOR='#000000' BORDER=0 CELLPADDING=5 CELLSPACING=0
WIDTH=500>
<TR><TD><CENTER>
<FONT FACE='Verdana,Arial' SIZE=+1 COLOR='#FFFFFF'><B>
Ticketing &amp; Reservation System
</B></FONT>
</CENTER>
</TD></TR>
</TABLE><p>
<FORM ACTION=reserve3.ihtml METHOD=POST>
<TABLE BGCOLOR='#000000' BORDER=0 CELLPADDING=0 CELLSPACING=0>
<TR><TD>
<TABLE BGCOLOR='#FFFFFF' BORDER=0 CELLPADDING=5
CELLSPACING=1>
    <TR>
    <TD COLSPAN=2><CENTER><FONT FACE='Verdana,Arial' SIZE=-
1><B>CUSTOMER INFORMATION</CENTER></TD>
    </TR><TR>
```

```
	<TD><FONT FACE='Verdana,Arial' SIZE=-1><B>Name as appear on
CC</B></FONT></TD>
	<TD><FONT FACE='Verdana,Arial' SIZE=-1><B><INPUT TYPE="text"
NAME="name" MAXLENGTH=50 SIZE=40></B></FONT></TD>
	</TR><TR>
	<TD><FONT FACE='Verdana,Arial' SIZE=-
1><B>Phone</B></FONT></TD>
	<TD><FONT FACE='Verdana,Arial' SIZE=-1><B><INPUT TYPE="text"
NAME="phone" MAXLENGTH=50 SIZE=12></B></FONT></TD>
	</TR><TR>
	<TD><FONT FACE='Verdana,Arial' SIZE=-
1><B>Email</B></FONT></TD>
	<TD><FONT FACE='Verdana,Arial' SIZE=-1><B><INPUT TYPE="text"
NAME="email" MAXLENGTH=50 SIZE=40></B></FONT></TD>
	</TR><TR>
	<TD><FONT FACE='Verdana,Arial' SIZE=-1><B>CC
Number</B></FONT></TD>
	<TD><FONT FACE='Verdana,Arial' SIZE=-1><B><INPUT TYPE="text"
NAME="ccnum" MAXLENGTH=50 SIZE=40></B></FONT></TD>
	</TR><TR>
	<TD><FONT FACE='Verdana,Arial' SIZE=-1><B>Expiration
(MM/YY)</B></FONT></TD>
	<TD><FONT FACE='Verdana,Arial' SIZE=-1><B>Month <INPUT
TYPE="text" NAME="ccmexp" MAXLENGTH=50 SIZE=2> / Year <INPUT
```

```
TYPE="text" NAME="ccyexp" MAXLENGTH=50 SIZE=2> ... example:
12/99</B></FONT></TD>
        </TR><TR>
        <TD COLSPAN=2>
        <CENTER>
        <FONT FACE='Verdana,Arial' SIZE=-1><B>
        <INPUT TYPE="submit" VALUE="Reserve Seats">
        <BR>
        <INPUT TYPE="reset" VALUE="Clear Choices">
        </B></FONT>
        </CENTER>
        </TD></TR>
</TABLE>
</TD></TR></TABLE>
<iEQ NAME="total" VALUE=0>
<iEQ NAME="date" VALUE='<iDATE>'>
<REM --- Begin database command to insert a new customer profile into the
database.  This step is primarily for the purpose of obtaining a new customer id
to associate this transaction.  The customer's ip and captured and inserted into a
new record in the "customers" table.  The marker, "new" is flagged "true" for the
attrieval of the new id.  Once the new customer id is captured, the marker "new"
is turned off.                                   --- >
<Markup language DBNAME=:datasource
        database="INSERT INTO customers (ip, new)
        VALUES (':i_ip',1)">
 <database DBNAME=:datasource
        database="SELECT id
        FROM customers
        WHERE new=1">
        <databaseFETCH>
        <iEQ NAME="custid" VALUE=:1>
</database>
<Markup language DBNAME=:datasource
        database="UPDATE customers
        SET new=0
        WHERE id=:custid">
<REM --- Begin database command to insert a new reservation into the database.
This step is primarily for the purpose of creating a new reservation record,
flagged with the newly obtained customer id, so that the transaction information
may be processed in "resere3.ihtml".                    --- >
<Markup language DBNAME=:datasource
        database="INSERT INTO orders (orderdate, approvalcode, receiptnum,
totalcharge, custid)
        VALUES (':date','0000', '0000', :total, :custid)"
FAILURE="The order could not be processed at this time due to technical
difficulties.">
<REM --- Begin database command to obtain the reservation id that was just
created. This information, coupled with the customer id, will be used to identify
```

XIX

```
this particular transaction in the database and be updated in "resere3.ihtml"

--->

<database DBNAME=:datasource

        database="SELECT max(id)

        FROM orders">

        <databaseFETCH>

        <iEQ NAME="oid" VALUE=:1>

</database>

        <INPUT TYPE="hidden" NAME="reserve" VALUE="receipt">

        <INPUT TYPE="hidden" NAME=custid VALUE=:custid>

        <INPUT TYPE="hidden" NAME=phone VALUE=:phone>

        <INPUT TYPE="hidden" NAME=email VALUE=:email>

        <INPUT TYPE="hidden" NAME=oid VALUE=:oid>

<Markup language DBNAME=:datasource database='INSERT INTO basket

(custid, pid, qty) VALUES (:custid, 6431, 1)'><database DBNAME=:datasource

database='SELECT cost FROM products WHERE

id=6431'><databaseFETCH><iEQ NAME='ticketprice'

VALUE=:1></database><iEQ NAME='total' VALUE=<iEVAL

EXPR=':ticketprice + :total' PREC='2'>><Markup language

DBNAME=:datasource database='INSERT INTO orderdetail (pid, oid, qty, sell)

VALUES (6431, :oid, 1, :ticketprice)'><Markup language DBNAME=:datasource

database='INSERT INTO basket (custid, pid, qty) VALUES (:custid, 6428,

1)'><database DBNAME=:datasource database='SELECT cost FROM products

WHERE id=6428'><databaseFETCH><iEQ NAME='ticketprice'
```

```
VALUE=:1></database><iEQ NAME='total' VALUE=<iEVAL
EXPR=':ticketprice + :total' PREC='2'>><Markup language
DBNAME=:datasource database='INSERT INTO orderdetail (pid, oid, qty, sell)
VALUES (6428, :oid, 1, :ticketprice)'><Markup language DBNAME=:datasource
database='INSERT INTO basket (custid, pid, qty) VALUES (:custid, 6429,
1)'><database DBNAME=:datasource database='SELECT cost FROM products
WHERE id=6429'><databaseFETCH><iEQ NAME='ticketprice'
VALUE=:1></database><iEQ NAME='total' VALUE=<iEVAL
EXPR=':ticketprice + :total' PREC='2'>><Markup language
DBNAME=:datasource database='INSERT INTO orderdetail (pid, oid, qty, sell)
VALUES (6429, :oid, 1, :ticketprice)'><Markup language DBNAME=:datasource
database='INSERT INTO basket (custid, pid, qty) VALUES (:custid, 6430,
1)'><database DBNAME=:datasource database='SELECT cost FROM products
WHERE id=6430'><databaseFETCH><iEQ NAME='ticketprice'
VALUE=:1></database><iEQ NAME='total' VALUE=<iEVAL
EXPR=':ticketprice + :total' PREC='2'>><Markup language
DBNAME=:datasource database='INSERT INTO orderdetail (pid, oid, qty, sell)
VALUES (6430, :oid, 1, :ticketprice)'><INPUT TYPE='hidden' NAME='total'
VALUE=:total></FORM></CENTER><P></BODY></markup language>
```

takes in credit card information and passes to:

```
<!Markup language>
<REM --- Imports the file "datasource.inc" which creates the variable
"datasource" which is used to tell Markup language which ODBC datasource to
connect to.
```

XXI

```
<iINCLUDE NAME="database\datasource.inc">

<REM -- In case a database or other type of error occurs, this will display the error

message. -- >

<ERROR>

<FONT FACE="Verdana,Arial" SIZE="+1"><B>An Error Has

Occurred</B></FONT><P>

<FONT FACE="Verdana,Arial" SIZE="-1"><B>Error Message =

:i_errortext</B></FONT><P>

<FONT FACE="Verdana,Arial" SIZE="-1"><B>database Error =

:i_databaseerrortext</B></FONT><P>

<FONT FACE="Verdana,Arial" SIZE="-1"><B>database Error =

:i_databaseerrorstmt</B></FONT><P>

</ERROR>

<REM --- Begin normal markup language here                    --- >

<markup language>

<HEAD>

<TITLE>Ticketing &amp; Reservation System - Confirmation</TITLE>

</HEAD>

<BODY BACKGROUND="images/background.jpg" TEXT="#000000"

LINK="#006666" VLINK="#006666">

<REM --- Checks to see if the instance "reserve" with a value of     --- >

<REM --- "receipt" is passed to this page.  If so, the codes within  --- >

<REM --- the iCASE and /iCASE tags are executed.  This is primarily  --- >

<REM --- for security purposes as the instanced "reserved" is only   --- >
```

```
<REM --- called in the step immediately before.  If a visitor        --- >
<REM --- accidentally stumbled upon this file, nothing will be run   --- >
<REM --- unless all previous steps are completed.                    --- >
<iCASE ALIAS="reserve" VALUE="receipt">
<CENTER>
<TABLE BGCOLOR='#000000' BORDER=0 CELLPADDING=5 CELLSPACING=0
WIDTH=500>
<TR><TD>
<CENTER>
<FONT FACE='Verdana,Arial' SIZE=+1 COLOR='#FFFFFF'><B>
Concert Reservation System
</B></FONT>
</CENTER>
</TD></TR>
</TABLE>
</CENTER><p>
<REM --- Two steps are combined here:                          --- >
<REM --- 1.  The reservation information captured in                --- >
<REM ---     "reserve2.ihtml" is passed to the iPAY tag which       --- >
<REM ---     processes the information and passes it to CyberCash  --- >
<REM ---     for verification.  If successful, the tag returns     --- >
<REM ---     "success", otherwise it returns a variation of "fail"  --- >
<REM --- 2.  iIF checks to see if the verification is a success or  --- >
<REM ---     failure.  If successful, it will process the codes    --- >
```

```
<REM ---    immediately after it, which includes updating the     --- >
<REM ---    database with a successful reservation.  If anything  --- >
<REM ---    other than "success" is returned, iIF will skip to    --- >
<REM ---    the iELSE tag and execute all codes thereafter.       --- >
<iIF COND=<iPAY SERVER="C3"
            amount=:total
            id=:oid
            ccnum=":ccnum"
            ccmexp=":ccmexp"
            ccyexp=":ccyexp"
            name=":name"
            capture="false"
            HOST="http://cr.cybercash.com/cgi-bin"
            PORT=80
      SECRET="vendorid-26"
            CRYPTOKEY="j1y1o1ohNU1ciTdPF1hsvHFjIpnCpR">>
<CENTER>
<TABLE BGCOLOR='#000000' BORDER=0 CELLPADDING=0 CELLSPACING=0>
<TR><TD>
<TABLE BGCOLOR='#FFFFFF' BORDER=0 CELLPADDING=5 CELLSPACING=1
WIDTH=500>
<TR><TD>
<FONT FACE="Verdana,Arial" SIZE="-1">
```

```
<B>The credit card has been approved and the reservation has been
processed.<P>
The following are the Authorization Code and Receipt Number:</B><p>
<b>Authorization Code:</b> :i_pay_authnumber<BR>
<b>Receipt Number:</b> :i_pay_transactionnumber<P>
<b>Customer Name:</b> :name<BR>
<b>Total Amount:</b> $:total<BR>
<b>Credit Card Number:</b> :ccnum<BR>
<b>Month of Expiration:</b> :ccmexp<BR>
<b>Year of Expiration:</b> :ccyexp<BR>
<REM --- Begin database query to update the "customers" table with customer
information captured in "reserve2.ihtml".          --- >
<Markup language DBNAME=:datasource
            database="UPDATE customers
            SET contact=':name', phone=':phone', email=':email'
            WHERE id=:custid">
<REM --- Begin database query to update the "orders" table is updated with the
approval code returned by CyberCash, as well as the transaction information
(total charge, credit card number, etc).                        --- >
<Markup language DBNAME=:datasource
            database="UPDATE orders
            SET approvalcode=':i_pay_authnumber',
receiptnum=':i_pay_transactionnumber', totalcharge=:total,cc=':ccnum',
ccm=':ccmexp', ccy=':ccyexp'
```

```
                WHERE id=:oid"

        FAILURE="The information was not saved correctly.<br>">

<REM --- Begin database query to turn off the availability of the seats that have

been reserved by setting the "active" attribute of the seats to "0".

--- >

<database ALIAS="markoff" dbname=":datasource" database="SELECT pid

FROM orderdetail WHERE oid=:oid">

        <databaseFETCH ALIAS="markoff">

        <iWHILE NOTALIAS=i_databaseempty>

                <Markup language dbname=":datasource"

                        database="UPDATE products

                                SET active=0

                                WHERE id=:markoff_pid">

                        <databaseFETCH ALIAS="markoff">

                </iWHILE ALIAS=i_databaseempty>

</database ALIAS="markoff">

</FONT>

</TD></TR>

</TABLE>

</TD></TR>

</TABLE>

</CENTER><p>

<iELSE>

<CENTER>
```

```
<TABLE BGCOLOR='#000000' BORDER=0 CELLPADDING=0 CELLSPACING=0>
<TR><TD>
<TABLE BGCOLOR='#FFFFFF' BORDER=0 CELLPADDING=5 CELLSPACING=1
WIDTH=500>
<TR><TD>
<FONT FACE="Verdana,Arial" SIZE="-1"><B>
This transaction could not be processed by Cyber Cash at this time. Either the
Credit Card Information was invalid or the Cyber Cash server is currently not
available.
</B></FONT>
</TD></TR>
</TABLE>
</TD></TR>
</TABLE>
</CENTER>
</iIF>
<FORM ACTION="index.ihtml" METHOD="post">
<CENTER>
<TABLE BGCOLOR='#000000' BORDER=0 CELLPADDING=0 CELLSPACING=0>
<TR><TD>
<TABLE BGCOLOR='#FFFFFF' BORDER=0 CELLPADDING=5 CELLSPACING=1
WIDTH=500>
<TR><TD>
<CENTER>
```

```
<FONT FACE="Verdana,Arial" SIZE="-1"><B>
<INPUT TYPE="submit" VALUE="Return to Concert Selection">
</B></FONT>
</CENTER>
</TD></TR>
</TABLE>
</TD></TR>
</TABLE>
</CENTER>
</FORM>
<FORM ACTION="http://domainname/filename" METHOD="post">
<CENTER>
<TABLE BGCOLOR='#000000' BORDER=0 CELLPADDING=0 CELLSPACING=0>
<TR><TD>
<TABLE BGCOLOR='#FFFFFF' BORDER=0 CELLPADDING=5 CELLSPACING=1
WIDTH=500>
<TR><TD>
<CENTER>
<FONT FACE="Verdana,Arial" SIZE="-1"><B>
<INPUT TYPE="submit" VALUE="Return to On-Line Ticketing & Reservation
Front Desk">
</B></FONT></CENTER>
</TD></TR></TABLE></TD></TR></TABLE></CENTER>
</FORM>
```

```
</iCASE ALIAS="reserve">

</BODY>

</markup language>
```

which verifies and confirms payment information

End Figure 5

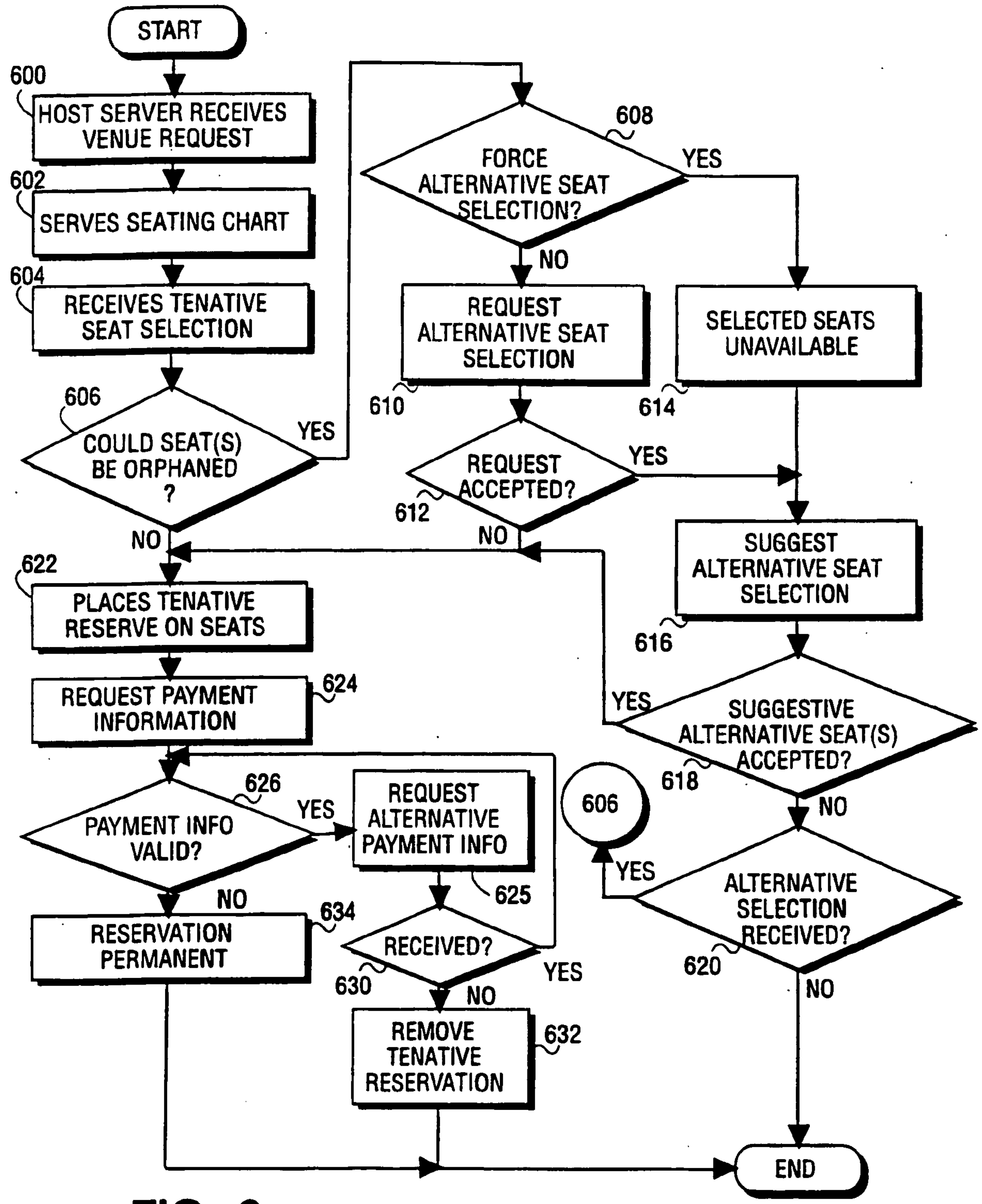
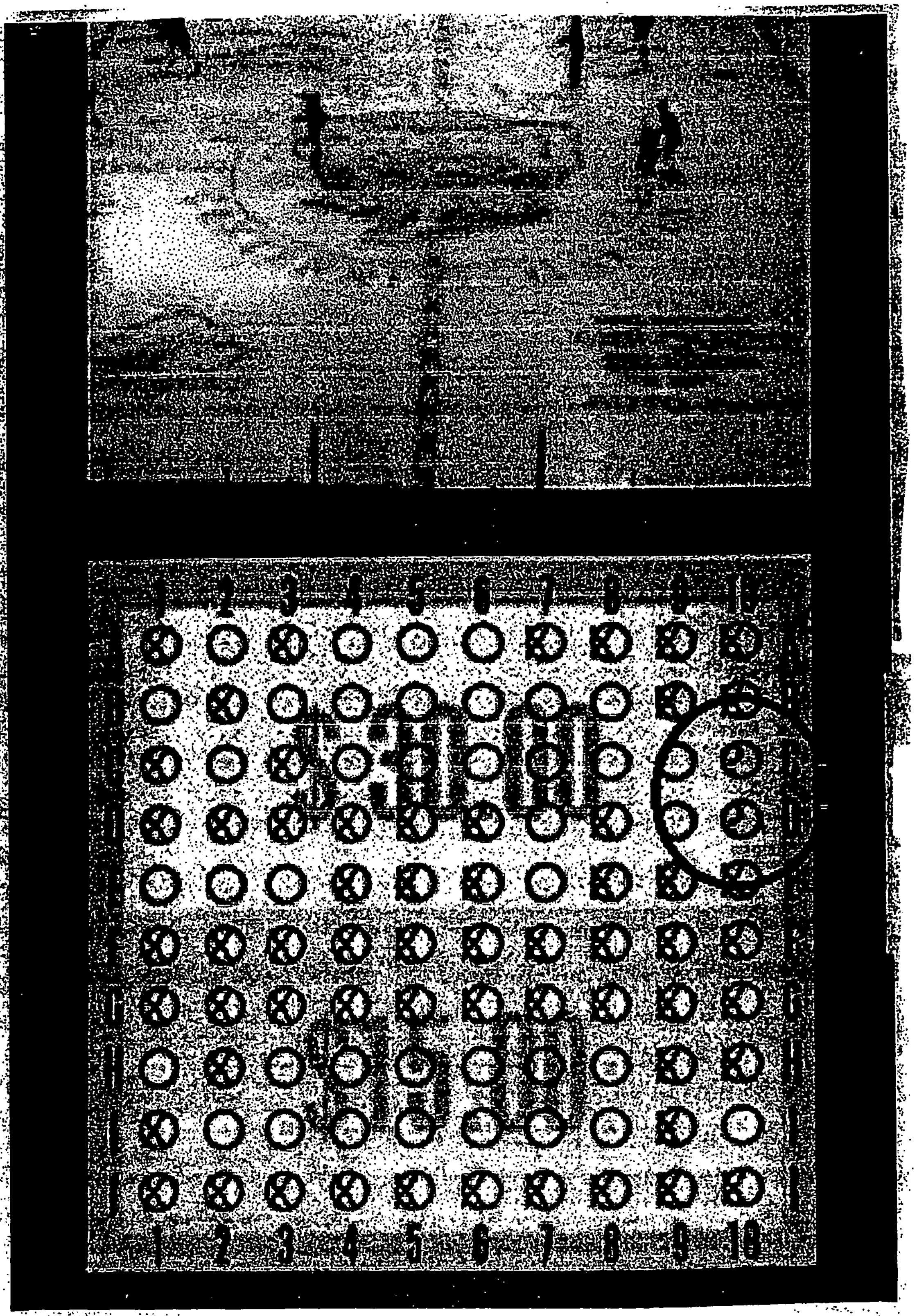

START
600
HOST SERVER RECEIVES VENUE REQUEST
602
SERVES SEATING CHART
604
RECEIVES TENATIVE SEAT SELECTION
606
COULD SEAT(S) BE ORPHANED ?
YES
NO
608
FORCE ALTERNATIVE SEAT SELECTION?
YES
NO
REQUEST ALTERNATIVE SEAT SELECTION
610
SELECTED SEATS UNAVAILABLE
614
REQUEST ACCEPTED?
612
YES
NO
SUGGEST ALTERNATIVE SEAT SELECTION
616
SUGGESTIVE ALTERNATIVE SEAT(S) ACCEPTED?
618
YES
NO
622
PLACES TENATIVE RESERVE ON SEATS
REQUEST PAYMENT INFORMATION
624
PAYMENT INFO VALID?
626
YES
NO
REQUEST ALTERNATIVE PAYMENT INFO
625
606
ALTERNATIVE SELECTION RECEIVED?
620
YES
NO
RESERVATION PERMANENT
634
RECEIVED?
630
YES
NO
REMOVE TENATIVE RESERVATION
632
END

FIG. 6


1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10

```
<!--- Begin Code // --- >

<!--- Defines the underlying image to lay the image map code over // --- >

<iIMAGEFROMFILE NAME="section" FILENAME ="images/map/sections/: b:
imagetype" TYPE="jpeg">

<! --- Defines the image source as a variable //--- >

<MAP NAME="section">

<! --- Opens ODBC connection // --- >

<iSQL DBNAME=":datasource"
      SQL="SELECT id, seatSection, seatRow, seatNumber, seatPrice,
seatAvailability, seatTemporary, x, y, shape, coords
      FROM seats
      WHERE seatSection=':b'"
      ALIAS="seats">

<!--- Performs SQL loop // --- >

      <iSQLFETCH ALIAS="seats">
            <iWHILE NOTALIAS=i_sqlempty>

<! --- Defines and populates seat availability and tempoary status //--- >

      <iEQ NAME="available" VALUE=:seats_seatAvailability>
      <iEQ NAME="temporary" VALUE=: seats_seatTemporary>

<! --- Begin conditions for temporary "?", reserved "X" or open seating
//--- >

<!--- Defines the condition if a seat is available // --- >

            <iCASE ALIAS="available" VALUE=1>

<!--- Defines the condition if a seat is currently being reserved // --->

            <iCASE ALIAS="temporary" VALUE=1>


<!--- Instructs iHTML to put a "?" over the coordinates retrieved from the
DB // --- >

                <iIMAGETEXT NAME="section" text="?" x=:seats_x
y=:seats_y color="red">
   </iCASE ALIAS="temporary">
```

```
<!--- Defines the condition if a seat is not currently being reserved // --->

                <iCASE ALIAS="temporary" VALUE=0>
                   <iEQ NAME="seatPrice" VALUE=<iEVAL
EXPR=" :seats_seatPrice" PREC=2>>

<!--- Instructs iHTML to write the image map coordinates retrieved from the DB to
make the area clickable // --- >

                   <AREA SHAPE=":seats _ shape" COORDS=": seats_coords"
HREF="angelsReserveList.ihtml?a=:a&seatID=:seats_id&seatSection=:seats_seatSection
&seatRow=:seats_seatRow&seatNumber=:seats_seatNumber&seatPrice=:
seats_seatPrice&option =add" TARGET="List" ALT="Section #:seats_seatSection,
Seat#:seats_seatRow-:seats_seatNumber" OnMouseOut="window. status="; return
true" OnMouseOver="window.status='Reserve Seat
:_seats_seatRow-:seats_ seatNumber of Section:seats_seatSection--$
:seatPrice'; return true">
                </iCASE ALIAS="temporary">

        </iCASE ALIAS="available">

<!--- Defines the condition if a seat is NOT available // --- >

        <iCASE ALIAS="available" VALUE=O>

<!--- Instructs iHTML to put a "X" over the coordinates retrieved from the
DB // --- >

                <iIMAGETEXT NAME="section" text="X" x=:_seats_x
y=:seats_y color="red">
             </iCASE ALIAS="available">

<!--- Ends SQL loop // --- >

  <iSQLFETCH ALIAS="seats">
</iWHILE ALIAS= i_sqlempty>

<!--- Closes ODBC connection //--- >

</iSQL ALIAS="seats">

</MAP>

 <!--- Writes the final image into a physical file //--->


<iIMAGEWRITE NAME="section"
FILENAME="images/map/sections/section-:b-:imagenum:imagetype" TYPE="jpeg"
QUALITY="25">
```

```
<!--- Removes the image source from memory //--- >

<iIMAGEDESTROY NAME="section">

<!--- End Code //--- >
```